Notes for Year 2 Gravitational-wave data analysis project

Ilya Mandel

(Dated: December 17, 2012; based on exercise prepared for Birmingham Gravitational-wave summer school, June 2012)

I. GENERAL COMMENTS

The goal of this project will be to simulate data analysis of gravitational-wave signals observable by detectors such as LIGO. In the process, you will learn data-analysis tools that are useful far beyond gravitational-wave astrophysics, including Fast Fourier Transforms, Wiener optimal matched filtering, and Bayesian analysis techniques such as Markov-chain Monte Carlo.

Software requirements: You will need to be able to write and compile code for your preferred language. You will also need to be able to make simple plots to visualize your results.

Preparation:

- Take a time-domain data set consisting of a single sinusoidal function, contained in the file *Sinusoidal.dat* (first column: time in seconds; second column: data set in dimensionless units). Plot the data. Use an FFT to determine the frequency of the sinusoidal. Check Parceval's theorem: is the power in the time-domain representation the same as in the frequency-domain representation?
- Take a time-domain data set consisting of a single sinusoidal function superimposed on white noise, contained in the file *NoisySinusoidal.dat*, organized as above. Plot the data in the time domain: can you see the sinusoidal signal buried in the noise? Now take an FFT and plot the data in the frequency domain. Estimate the frequency of the sinusoidal from the latter.
- Take the same data set as above. Carry out matched filtering against a signal of the form $h(t) = A\sin(2\pi ft)$ for various values of f. (In this case, since the noise is white, filtering just corresponds to taking the inner product of the data against the model signal.) Find which f yields a peak of the filter output and compare with the above.
- Take FFTs of short sinusoidal waveform with box windows to observe effects of windowing and spectral leakage.

From here on, you will work directly with time-domain data to avoid FFT artifacts.

You will be given three files. They will consist of (1) the noise power spectral density of a single detector; (2) a basic data set containing frequency-domain data recorded by that detector, for which all parameters but the chirp mass are known [in particular, $t_c = 0$s, $\phi_c = 0$]; (3) a challenge data set containing the frequency-domain data recorded by that detector, with four unknown parameters: chirp mass, distance, time of coalescence and phase at coalescence. Over the course of the project, you will develop the software to compute frequency-domain templates, likelihood/overlap functions, and the tools for carrying out a grid-based matched filtering search and a Markov Chain Monte Carlo to decode the signal parameters.

Recommended steps:

- Read from the input files, code up the frequency-domain 0pN waveform, plot the waveform and noise PSD.
- Implement grid-based matched filtering in one dimension for the basic data set to determine the best-fit mass parameter; challenge do same for the challenge data set, maximizing over time and phase with inverse FFTs.
- Implement likelihood function and a core Markov Chain Monte Carlo engine (MCMC).
- Deploy MCMC on basic data set to get a posterior probability density function on the distance (fixing the chirp mass to the best-fit value found previously); challenge apply MCMC to the challenge data set, performing a four-dimensional MCMC in chirp mass, time and phase at coalescence, and distance parameter space.
- Compare results and boast of the fruits of your labor.

Some technical details follow.

II. NOTATION, SNR, LIKELIHOOD, BAYES' THEOREM

The overlap of two signals is defined as the inner product

$$< a|b >= 4\Re \int_0^\infty \frac{a(f)b^*(f)}{S_n(|f|)} df, \tag{1}$$

where $S_n(|f|)$ is the one-sided noise power spectral density.

Below, waveforms are denoted by $h(\vec{\theta})$ (where $\vec{\theta}$ is the parameter vector), noise is n, and the signal consisting of a waveform with parameters $\vec{\theta}_{inj}$ injected into noise is $d = h(\vec{\theta}_{inj}) + n$.

The signal-to-noise ratio SNR is

$$\rho = \max_{\vec{\theta}} \frac{< d|h(\vec{\theta}) >}{\sqrt{< h(\vec{\theta})|h(\vec{\theta}) >}}. \tag{2}$$

The likelihood L for a given parameter vector $\vec{\theta}$ is

$$L(\vec{\theta}) = p(d|\vec{\theta}) = exp\left(-\frac{< d - h(\vec{\theta})|d - h(\vec{\theta}) >}{2}\right). \tag{3}$$

Bayes' theorem gives the posterior probability density function as the product of the likelihood and the prior, divided by the evidence (for normalization):

$$p(\vec{\theta}|d) = \frac{p(d|\vec{\theta})p(\vec{\theta})}{p(d)}. \tag{4}$$

III. WAVEFORMS

We will use a simple frequency-domain inspiral waveform (TaylorF2), computed at the zeroth post-Newtonian order in both amplitude, and cut off at the innermost stable circular orbit (ISCO).

Specifically, the waveform is composed of an amplitude and phase,

$$\tilde{h}(f) = A(f)e^{i\Psi(f)}, \tag{5}$$

where

$$A(f) = \frac{1}{d_L}\sqrt{5/24}\pi^{-2/3}M_c^{5/6}f^{-7/6}, \tag{6}$$

$$\Psi(f) = \frac{3}{128}(\pi M f)^{-5/3}\frac{1}{\eta} + 2\pi f t_c - \phi_c, \tag{7}$$

and M_c, t_c, ϕ_c and d_L are the chirp mass, time of coalescence, phase of coalescence, and luminosity distance. We will assume that all waveforms correspond to equal-mass systems, so the symmetric mass ratio η can be fixed to $\eta = 0.25$ for all exercises. The total mass M of the binary is given by $M = M_c/\eta^{3/5}$.

The amplitude of the waveform is set to zero for all frequencies $f > f_{\rm ISCO}$, where

$$f_{\rm ISCO} = \frac{1}{\pi 6^{1.5}}\frac{1}{M} \tag{8}$$

is the frequency of gravitational waves at the ISCO.

IV. MCMC

The Metropolis-Hastings algorithm works as follows. When in state $\vec{\theta}_i$, we choose a trial state $\vec{\theta}'$ based on some "trial jump" probability distribution $Q(\vec{\theta}_i \to \vec{\theta}')$. We are free to choose the trial jump probability distribution, so long as it is possible to completely explore parameter space with trial jumps. We accept the trial state $\vec{\theta}'$ with a probability given by

$$R(\vec{\theta}_i \to \vec{\theta}') \equiv \min\left\{\frac{p(\vec{\theta}')Q(\vec{\theta}' \to \vec{\theta}_i)}{p(\vec{\theta}_i)Q(\vec{\theta}_i \to \vec{\theta}')},\ 1\right\} \tag{9}$$

If the proposed trial state is accepted, $\vec{\theta}_{i+1} = \vec{\theta}'$; otherwise, we set $\vec{\theta}_{i+1} = \vec{\theta}_i$.

The ratio of Q's above is needed to satisfy detailed balance. In the simple case when the jump proposal distribution is symmetric, the ratio of Q's is equal to 1.

Finding a suitable jump proposal is key to an efficient MCMC. One simple option could be a Gaussian distribution of $(\vec{\theta}' - \vec{\theta})$ with mean zero and standard deviation σ. The goal is to have a jump proposal that is sufficiently broad to explore the full parameter range quickly when burning in, yet narrow enough to have a high acceptance rate once the peak of the posterior is being sampled. One can either use a mix of narrow and wide jumps (e.g., use two different values of σ, choosing a large or a small σ for some fraction of jumps. Another alternative is adaptation, where σ is allowed to gradually evolve over the course of the MCMC, growing when jumps are being accepted, and being reduced when they are rejected. An overall acceptance ratio of around 25% is suitable in many applications.

Whether an MCMC has converged to the true posterior probability distribution is a tricky question. One can try several independent MCMC runs to check whether their sampled distributions match. For a single run, it is often useful to check that one has a sufficiently large number of uncorrelated points by downsampling by the autocorrelation length of the samples (neighboring samples are typically correlated because local jump proposals are used, or because jumps are being rejected). Several statistics are available to check for violations of convergence, though none can guarantee convergence.

V. INJECTIONS

We have two data sets available. The first set, *Data.dat*, contains a fairly high SNR signal with known $t_c = 0$, $\phi_c = 0$. Only the chirp mass and distance are unknown. The second data set is a challenge: *Challenge.dat* contains a lower-SNR signal with unknown chirp mass, time and phase of coalescence, and luminosity distance within the prior ranges specified above. The file *Noise.dat* contains the noise power spectral density, the same for both data sets.

The data set files have three columns: frequency (in Hz), the real part of the frequency-domain data $\Re(\tilde{d}(f))$, and the imaginary part of the frequency-domain data $\Im(\tilde{d}(f))$. The last two columns have units of s. The noise file has two columns: frequency (in Hz) and the noise power spectral density $S_n(f)$ in 1/Hz.

For this exercise, we will assume flat priors on the parameters. Unless fixed, these take the following values: $M_c \in [1, 10]$ solar masses; $t_c \in [-0.1, 0.1]$s; $\phi_c \in [0, 2\pi]$; $d_L \in [1, 2000]$ Mpc.

Hint for the basic data set: the peak around the true chirp mass value is quite narrow, and is easy to miss if your grid is too coarse. If all else fails, try a grid in chirp values that is an exact multiple of $0.1M_\odot$: 1.0, 1.1, 1.2, ... solar masses.

VI. CONSTANTS

The expressions above are written in geometrical units, $G = c = 1$. However, since the frequency is given in Hz and the noise PSD in 1/Hz, it is convenient to convert masses and distances to units of time:

A solar mass, $M_\odot$, is equal to 4.92674×10^{-6}s.

A megaparsec, Mpc, is equal to 1.029×10^{14}s.